

5 minute read
 ✓ page test

HTTPS Proxy

The Configure an Egress Gateway example shows how to direct traffic to external services from your mesh via an Istio edge component called *Egress Gateway*. However, some cases require an external, legacy (non-Istio) HTTPS proxy to access external services.

proxy in place and all the applications within the organization may be required to direct their traffic through it.

For example, your company may already have such a

external HTTPS proxy. Since applications use the HTTP CONNECT method to establish connections with HTTPS proxies, configuring traffic to an external HTTPS proxy is different from configuring traffic to external HTTP and HTTPS services.

This example shows how to enable access to an

Before you begin

• Setup Istio by following the instructions in the Installation guide.

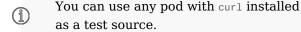


 Deploy the sleep sample app to use as a test source for sending requests. If you have automatic sidecar injection enabled, run the following command to deploy the sample app:

```
$ kubectl apply -f @samples/sleep/sleep.yaml@
```

Otherwise, manually inject the sidecar before deploying the sleep application with the following command:

```
$ kubectl apply -f <(istioctl kube-inject -f @samples/sleep
/sleep.yaml@)</pre>
```



• Set the SOURCE_POD environment variable to the name of your source pod:

```
$ export SOURCE_POD=$(kubectl get pod -l app=sleep -o jsonp
ath={.items..metadata.name})
```

Enable Envoy's access logging

Deploy an HTTPS proxy

To simulate a legacy proxy and only for this example,

you deploy an HTTPS proxy inside your cluster. Also, to simulate a more realistic proxy that is running outside of your cluster, you will address the proxy's pod by its IP address and not by the domain name of a Kubernetes service. This example uses Squid but you can use any HTTPS proxy that supports HTTP CONNECT. 1. Create a namespace for the HTTPS proxy, without labeling it for sidecar injection. Without the label, sidecar injection is disabled in the new namespace so Istio will not control the traffic there. You need this behavior to simulate the

proxy being outside of the cluster.

\$ kubectl create namespace external

```
2. Create a configuration file for the Squid proxy.
```

```
$ cat <<EOF > ./proxy.conf
http port 3128
acl SSL_ports port 443
acl CONNECT method CONNECT
http access denv CONNECT !SSL ports
http access allow localhost manager
http access deny manager
http access allow all
```

coredump_dir /var/spool/squid EOF

- 3. Create a Kubernetes ConfigMap to hold the configuration of the proxy:
 - om-file=squid.conf=./proxy.conf

\$ kubectl create configmap proxy-configmap -n external --fr

4. Deploy a container with Squid:

\$ kubectl apply -f - <<EOF

```
apiVersion: apps/v1
kind: Deployment
metadata:
   name: squid
   namespace: external
spec:
   replicas: 1
   selector:
    matchLabels:
```

```
app: squid
  template:
    metadata:
      labels:
        app: squid
    spec:
      volumes:
      - name: proxy-config
        configMap:
          name: proxy-configmap
      containers:
      - name: squid
        image: sameersbn/squid:3.5.27
        imagePullPolicy: IfNotPresent
        volumeMounts:
        - name: proxy-config
          mountPath: /etc/squid
          readOnly: true
FOF
```

to test traffic to the proxy without Istio traffic control.

\$ kubectl apply -n external -f @samples/sleep/sleep.yaml@

6. Obtain the IP address of the proxy pod and define

5. Deploy the sleep sample in the external namespace

- the PROXY_IP environment variable to store it:

 \$ export PROXY_IP="\$(kubectl get pod -n external -l app=squ id -o jsonpath={.items..podIP})"
- 7. Define the PROXY_PORT environment variable to store the port of your proxy. In this case, Squid uses port 3128.

\$ export PROXY PORT=3128

```
$ kubectl exec "$(kubectl get pod -n external -1 app=sleep
-o jsonpath={.items..metadata.name})" -n external -- sh -c
"HTTPS_PROXY=$PROXY_IP:$PROXY_PORT curl https://en.wikipedi
a.org/wiki/Main_Page" | grep -o "<title>.*</title>"
<title>Wikipedia, the free encyclopedia</title>
```

9. Check the access log of the proxy for your request:

\$ kubectl exec "\$(kubectl get pod -n external -l app=squid

• You deployed the HTTPS proxy.

Istio:

So far, you completed the following tasks without

- You used curl to access the wikipedia.org external
- service through the proxy.

Next, you must configure the traffic from the Istioenabled pods to use the HTTPS proxy.

Configure traffic to external HTTPS proxy

 Define a TCP (not HTTP!) Service Entry for the HTTPS proxy. Although applications use the HTTP CONNECT method to establish connections with HTTPS proxies, you must configure the proxy for TCP traffic, instead of HTTP. Once the connection is established, the proxy simply acts as a TCP tunnel.

```
hosts:
       - my-company-proxy.com # ignored
      addresses:
       - $PROXY IP/32
      ports:
       - number: $PROXY PORT
        name: tcp
        protocol: TCP
      location: MESH EXTERNAL
    FOF
2. Send a request from the sleep pod in the default
   namespace. Because the sleep pod has a sidecar,
```

\$ kubectl apply -f - <<EOF

kind: ServiceEntry
metadata:
 name: proxy

spec:

apiVersion: networking.istio.io/v1beta1

=\$PROXY_IP:\$PROXY_PORT curl https://en.wikipedia.org/wiki/M ain_Page" | grep -o "<title>.*</title>" <title>Wikipedia, the free encyclopedia</title>

\$ kubectl exec "\$SOURCE POD" -c sleep -- sh -c "HTTPS PROXY

Istio controls its traffic.

3. Check the Istio sidecar proxy's logs for your request:
\$ kubectl logs "\$SOURCE POD" -c istio-proxy

```
$ kubect1 logs "$SOURCE_POD" -c istlo-proxy
[2018-12-07T10:38:02.841Z] "- - - " 0 - 702 87599 92 - "-" "
-" "-" "172.30.109.95:3128" outbound[3128]|my-company-p
roxy.com 172.30.230.52:44478 172.30.109.95:3128 172.30.230.
52:44476 -
```

4. Check the access log of the proxy for your

request:

Understanding what happened

In this example, you took the following steps:

proxy.2. Created a TCP service entry to enable Istiocontrolled traffic to the external proxy.

1. Deployed an HTTPS proxy to simulate an external

Note that you must not create service entries for the external services you access through the external proxy, like wikipedia.org. This is because from Istio's point of view the requests are sent to the external proxy only; Istio is not aware of the fact that the external proxy forwards the requests further.

Cleanup

namespace:

1. Shutdown the sleep service:

```
$ kubectl delete -f @samples/sleep/sleep.yaml@
```

2. Shutdown the sleep service in the external

\$ kubectl delete -f @samples/sleep/sleep.yaml@ -n external

```
3. Shutdown the Squid proxy, remove the {\tt ConfigMap} and the configuration file:
```

```
$ rm ./proxy.conf4. Delete the external namespace:
```

\$ kubectl delete -n external deployment squid
\$ kubectl delete -n external configmap proxy-configmap

5. Delete the Service Entry:

\$ kubectl delete namespace external

\$ kubectl delete serviceentry proxy